Computation of Posterior Marginals on Aggregated State Models for Soft Source Decoding

Simon Malinowski, Hervé Jégou, Christine Guillemot

Computation of posterior marginals on aggregated state models for soft source decoding

Simon Malinowski
IRISA/University of Rennes

Hervé Jégou
INRIA

Christine Guillemot
IRISA/INRIA

***Abstract*— Optimum soft decoding of sources compressed with variable length codes and quasi-arithmetic codes, transmitted over noisy channels, can be performed on a bit/symbol trellis. However, the number of states of the trellis is a quadratic function of the sequence length leading to a decoding complexity which is not tractable for practical applications. The decoding complexity can be significantly reduced by using an aggregated state model, while still achieving close to optimum performance in terms of bit error rate and frame error rate. However, symbol a posteriori probabilities can not be directly derived on these models and the symbol error rate (SER) may not be minimized. This paper describes a two-step decoding algorithm that achieves close to optimal decoding performance in terms of SER on aggregated state models. A performance and complexity analysis of the proposed algorithm is given.**

I. Introduction

Extensive research effort has been dedicated to the problem of soft decoding and joint source/channel decoding of sequences of symbols compressed with variable length codes (VLCs) [1]–[9] and transmitted over noisy channels. Given the received noisy bit-stream, the problem is to estimate the sequence of symbols which has been compressed and transmitted. The decoding process is usually modeled as an automaton, and a Bayesian estimation is then run on a trellis representation of this automaton. Two trellises have been considered to estimate the emitted sequence of symbols: the bit-level trellis proposed in [1] and the bit/symbol trellis [4]. The bit/symbol trellis coupled with the BCJR algorithm [10] allows the minimization of either the bit error rate (BER), or the symbol error rate (SER). The bit/symbol trellis allows in addition the incorporation of a termination constraint which guarantees that the number of decoded symbols is equal to the number of transmitted symbols. However, the number of states of the bit/symbol trellis is a quadratic function of the sequence length, and hence the decoding complexity on this trellis may not be tractable for typical sequence lengths.

Quasi-arithmetic (QA) codes, introduced in [11], have also recently retained the attention of researchers because of their use in practical compression systems such as JPEG-2000 or MPEG-4/AVC. In [5] and [12], methods are proposed to represent QA codes as state machines, which, in principle, may be of infinite size. Several methods exist however to keep the number of states finite [6] [12] [13], at the cost of a slight loss in terms of compression efficiency. An optimal state model for soft decoding of QA codes exploiting a termination constraint on the number of encoded symbols is presented in [5]. However, the decoding complexity associated with this model is high.

Aggregated state models allowing soft decoding with reduced complexity of sources encoded with VLCs [7] and with QA codes [8] have been proposed. Minimum frame error rates (FER) and BER can be obtained by running a Viterbi [14] or a BCJR algorithm, respectively, on these models. In contrast with the optimal models used in [4] and [5], respectively for VLC and QA codes, aggregated state models do not keep track of the symbol clock values. They only keep track of the symbol clock value modulo an integer T, which is denoted M_k for a given bit clock instant k. Thus, symbol A Posteriori Probabilities (APP) do not come as a natural product of the estimation with aggregated state models. Yet, symbol APP are required to minimize the SER, as well as to use the estimation algorithm in an iterative source-channel decoding structure (e.g. in [9]).

In this paper, we describe a low complexity method to compute symbol APP on aggregated state models for soft decoding of sources encoded either with VLCs or with QA codes. The decoding proceeds in two steps. In a first step, a Viterbi algorithm is run on the aggregated model with parameter T to select the sequence of aggregated states with the maximum a posteriori probability. The values T_k which can be taken by the symbol clock for a given value m_k of the symbol clock modulo T are then estimated. This defines a restricted set for the possible values of the symbol clock T_k at each instant k. In a second step, a BCJR is run on the aggregate state model whose states are assigned the estimated symbol clock values found in the first step. Symbol APP are thus naturally obtained as an output of the BCJR algorithm. Simulation results reveal that the SER obtained with the reduced complexity algorithm, as well as the a posteriori probabilities on each symbol, closely approach the ones obtained with the optimal state model.

The rest of this paper is organised as follows. The aggregated models of [7] and [8] are first recalled in Section II. The proposed decoding algorithm is then presented in Section III together with a complexity and decoding performance analysis.

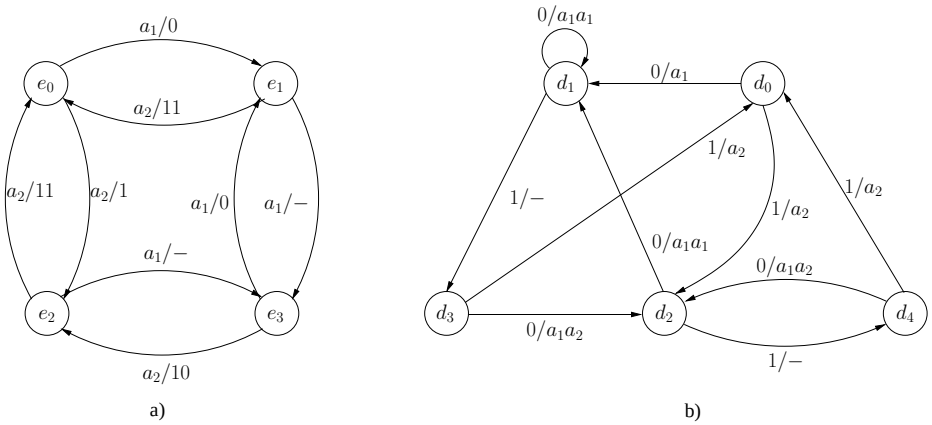

Fig. 1. Encoding (a) and decoding (b) FSM associated with code $\mathcal{Q}_1$.

II. Aggregated models for soft decoding of VLC and QA codes

Let $\mathbf{U} = U_1, \ldots, U_t, \ldots, U_{L(\mathbf{U})}$ be a sequence of $L(\mathbf{U})$ symbols to be compressed and transmitted. These symbols take their values in an M-ary alphabet $\mathcal{A} = \{a_1, \ldots, a_M\}$. The sequence $\mathbf{U}$ is encoded using a VLC or a QA code leading to a bit-stream $\mathbf{X} = X_1, ...X_k, ...X_{L(\mathbf{X})}$ of variable length $L(\mathbf{X})$. Note that, in practical compression systems using QA codes, the M-ary source is first binarized, so that binary symbols (i.e. $\mathcal{A} = \{a_1, a_2\}$) are fed to the QA encoder. The resulting bit-stream is assumed here to be transmitted using a binary phase shift keying (BPSK) modulation over an additive white Gaussian noise (AWGN) channel. The channel is characterized by its signal to noise ratio, denoted E_b/N_0 and expressed in decibels (dB). The noisy observations at the output of the channel are denoted $\mathbf{Y} = Y_1, ..., Y_{L(\mathbf{X})}$. In the following, the notation $Y_t^{t'}$ will denote the sequence of observations $Y_t, \ldots, Y_{t'}$.

The method described here applies to soft decoding of both VLCs and QA codes. Both types of codes can be defined by finite state machines (FSM) [4] [5]. The general method to construct the FSM associated with a QA code is detailed in [6]. The encoding and decoding FSM of a QA code, denoted $\mathcal{Q}_1$ and defined on the initial interval $[0, 8[$ for an input binary source of probabilities ($\mathbb{P}(a_1) = 0.7$, $\mathbb{P}(a_2) = 0.3$), are depicted in Fig. 1.

The optimal model corresponding to the bit/symbol trellis is defined by pairs of random variables (N_k, T_k), where N_k represents the internal state of the VLC (i.e., the internal node of the VLC code tree) at the bit clock instant k or the internal state of the decoding automaton for the QA code [5]. The random variable T_k represents the symbol clock value at the bit clock instant k. In the following, $\mathcal{N}$ will denote the set of states of the decoding automaton, and card($\mathcal{N}$) its cardinal. This model keeps track of the symbol clock during the decoding process. A termination constraint forcing the number of symbols of the estimated sequence to be equal to $L(\mathbf{U})$ can thus be easily introduced on this state model. However, the number of states is a quadratic function of the sequence length (equivalently the bit-stream length). The computational cost may thus not be tractable for typical values of the sequence length $L(\mathbf{U})$.

Aggregated state models have been introduced in [7] and [8] for soft decoding of VLC and QA codes, respectively. They are defined by pairs of random variables $s_k = (N_k, M_k)$, where N_k represents the state of the decoder at the bit clock instant k, and $M_k = T_k \mod T$ is the remainder of the Euclidean division of T_k by T. The transitions that correspond to the decoding of σ symbol(s) modify M_k as $M_k = (M_{k-1} + \sigma) \mod T$. Hence, the transition probabilities on the aggregated state models are given by

$$\mathbb{P}(N_k = n_k, M_k = m_k \,|\, N_{k-1} = n_{k-1}, M_{k-1} = m_{k-1}) = \begin{cases} \mathbb{P}(N_k = n_k \,|\, N_{k-1} = n_{k-1}) & \text{if } m_k = (m_{k-1} + \sigma) \bmod T \\ 0 & \text{otherwise,} \end{cases} \tag{1}$$

where the probabilities $\mathbb{P}(N_k = n_k \,|\, N_{k-1} = n_{k-1})$ are deduced from the source statistics. The parameter T controls the trade-off between estimation accuracy and decoding complexity. The termination constraint on this model amounts to forcing the number of symbols modulo T of the estimated sequence to be equal to $m_{L(\mathbf{X})} = L(\mathbf{U}) \mod T$. Although this constraint is weaker than the one available on the optimal trellis, it is shown in [7] that close to optimal decoding performance are obtained with these models provided that the parameter T is above a given value T^* which depends on the re-synchronization properties of the codes. A method to estimate the value of T^* is given in [7].

Minimum FER and BER can be obtained by running the Viterbi and the BCJR algorithm, respectively, on these aggregated state models. However, since the aggregated model with parameter T keeps track "only" of symbol clock values modulo T, and not of the actual symbol clock values as in the optimal model, symbol APPs, which may be needed when using the decoder in an iterative structure, are not anymore a natural product of the estimation. In addition, the SER may not be minimized. In the next section, an algorithm allowing

the computation of symbol APPs on the aggregated model, as well as the decoding with close to optimum performance, is described.

III. Computation of Symbol Posterior Marginals

A. Description of the proposed algorithm

Let us consider the aggregated state model with parameter T_η^*, where $\eta \in [0,1]$. The parameter T_η^* is the smallest value of the aggregation parameter T such that the Viterbi algorithm run on the model with parameter T_η^* will select a symbol sequence of length $L(\mathbf{U})$ (that is, with the right number of symbols) with a probability higher than or equal to $1-\eta$. The value T_η^* depends on the channel signal to noise ratio E_b/N_0, on the sequence length $L(\mathbf{U})$ and on the code resynchronization properties. The computation of T_η^* is detailed in [7].

The symbol posterior marginals can be directly derived from the APP of pairs (N_k, T_k). These quantities are however not naturally available on the aggregated state model. Running the BCJR algorithm on the aggregated state model will only produce APP of pairs (N_k, M_k). To derive the APP of (N_k, T_k) from the APP of (N_k, M_k), one has to estimate the values T_k that the symbol clock can take for a given value m_k of the symbol clock modulo T_η^* (that is for $M_k = m_k$). A BCJR can then be run on the aggregated model, using the estimated symbol clock values to compute the symbol APP. More precisely, the algorithm proceeds in two steps as follows:
Step 1 - Symbol clock mapping : A Viterbi algorithm run on the trellis of parameter T_η^* selects a sequence of states $\{s_1^* = (n_1^*, m_1^*), \ldots, s_{L(X)}^* = (n_{L(X)}^*, m_{L(X)}^*)\}$ maximizing the APP $\mathbb{P}(s_1 = s_1^*, \ldots, s_{L(X)} = s_{L(X)}^*) \,|\, Y_1^{L(X)})$. The quantity m_k^* denotes the value taken by the random variable M_k in the sequence selected by the Viterbi algorithm (i.e., the most probable sequence). The above sequence of states corresponds to an estimated sequence of bits $\hat{X} = \hat{X}_1 \ldots \hat{X}_{L(X)}$. Hard decoding of $\hat{X}$ gives an estimate of the symbol sequence $\hat{U} = \hat{U}_1 \ldots \hat{U}_{L(U)}$, from which can be directly obtained an estimate of the symbol clock values $\hat{t}_k$ corresponding to each value m_k^*.

The knowledge of the estimate of the symbol clock values corresponding to each m_k^* is not sufficient to compute symbol APPs on the aggregated model. This APP can be obtained from (N_k, T_k) by estimating for each value m_k (and not only m_k^*), the associated estimate of the symbol clock. For that purpose, we define for each modulo value m_k, the value $\varphi(\hat{t}_k, m_k)$ of the symbol clock which is the closest to $\hat{t}_k$ and congruent to m_k modulo T_η^*:

$$\varphi(\hat{t}_k, m_k) = \hat{t}_k + \left\lfloor \tfrac{T_\eta^*-1}{2} \right\rfloor + \left((m_k - m_k^* + \left\lfloor \tfrac{T_\eta^*-1}{2} \right\rfloor) \bmod T_\eta^* \right), \quad (2)$$

where the term $\left((m_k - m_k^* + \left\lfloor \frac{T_\eta^*-1}{2} \right\rfloor) \bmod T_\eta^* \right)$ is forced to be in $\{0, \ldots, T_\eta^* - 1\}$. Note that when $m_k = m_k^*$, $\varphi(\hat{t}_k, m_k) = \hat{t}_k$. In other words, at each bit clock instant k, a subset of T_η^* consecutive symbol clock values, centered on, $\hat{t}_k$ is selected according to (2).

Step 2 - APP computation : A BCJR algorithm is then applied on the aggregated state model defined by the pairs of variables $\mathbf{s}_k = (N_k, M_k)$, on which each state s_k is assigned the estimated symbol clock value $\varphi(\hat{t}_k, M_k)$ corresponding to the values obtained in Step 1 for each state of the aggregated model. For all $n \in \{1, \ldots, \text{card}(\mathcal{N})\}$, and for all $m \in \{0, \ldots, T_\eta^* - 1\}$, the forward and backward passes of the BCJR algorithm compute, for each state $v = (n, m)$ of the trellis, the probability functions [10]

$$\alpha_k(v) = \mathbb{P}(\mathcal{V}_k = v; \mathbf{Y}_1^k) \quad (3)$$

$$\beta_k(v) = \mathbb{P}(\mathbf{Y}_{k+1}^{L(\mathbf{X})} \,|\, \mathcal{V}_k = v), \quad (4)$$

for $1 \le k \le L(\mathbf{X})$. Let us define, for every symbol a_i of the alphabet, the set $\tau_j(a_i)$ of state pairs $(v = (n, m), v' = (n', m'))$ in the decoding trellis such that the j^{th} symbol output by the transition from v to v' is equal to a_i. This set of state pairs is directly obtained from the decoding automaton of the QA code or from the VLC code tree. Note that for VLC, the sets $\tau_j(a_i)$ for $j > 1$ are empty. Then, the symbol APP on the trellis defined by the states $\mathcal{V}_k$ is obtained by:

$$\mathbb{P}(U_t = a_i \,|\, Y_1^{L(\mathbf{X})}) \propto \sum_{1 \le k \le L(\mathbf{X})} \sum_{j \ge 1} \sum_{(v,v') \in \Omega_j^t(a_i)} \alpha_k(v)\, \beta_k(v')\, \gamma_k(v, v'), \quad (5)$$

where $\gamma_k(v, v') = \mathbb{P}(\mathcal{V}_k = v'; Y_k \,|\, \mathcal{V}_{k-1} = v)$ is calculated from the channel measures and the source model, and where $\Omega_j^t(a_i)$ is the set of state pairs (v, v') in the decoding trellis such that $(v, v') \in \tau_j(a_i)$ and $\varphi(\hat{t}_k, m) = t - j$.

Note that, for sake of clarity, the two steps of the algorithm are described separately. However, for implementation purposes, the Viterbi algorithm (Step 1) and the forward pass of the BCJR algorithm can be done simultaneously. In that case, two different measures have to be stored for each state of the trellis at a bit clock instant k : $\max \mathbb{P}(V_1, \ldots, \mathcal{V}_k; Y_1^k)$ for the Viterbi algorithm and the probability in (3) for the BCJR algorithm. The computation of the branch metric γ can hence be done only once.

B. Complexity analysis

The number of states of the aggregated model with parameter T_η^* is linear with the parameter of the model and the sequence length (in bits) $L(\mathbf{X})$ and is about $T_\eta^* \times \text{card}(\mathcal{N}) \times L(\mathbf{X})$. The complexity $\mathcal{D}(T_\eta^*)$ of the algorithm on the model with parameter T_η^* is proportional to the number of states, hence $\mathcal{C}(T_\eta^*) = \gamma\, T_\eta^* \times \text{card}(\mathcal{N}) \times L(\mathbf{X})$, where γ is a constant. The complexity of the Viterbi algorithm on the same model is assumed to be lower than the one of the BCJR algorithm as only a forward pass is processed. Therefore, the complexity $\mathcal{D}_{pa}$ of the proposed algorithm verifies

$$\mathcal{D}_{pa} \le 2\, \gamma\, T_\eta^* \times \text{card}(\mathcal{N}) \times L(\mathbf{X}). \quad (6)$$

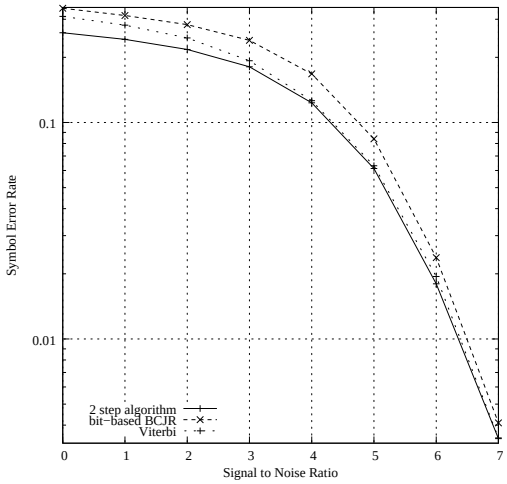

Fig. 2. SER versus signal to noise ratio for code $\mathcal{Q}_1$

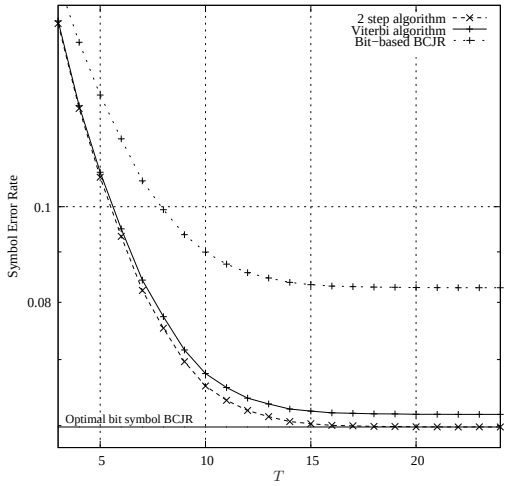

Fig. 3. SER versus the aggregation parameter for $E_b/N_0 = 5\,\text{dB}$ and for code $\mathcal{C}_1$

Thus, the proposed algorithm ensures a complexity reduction compared to a complete BCJR algorithm ($T = L(\mathbf{U})$) provided that $T_\eta^* \leq L(\mathbf{U})/2$. For the proposed VLC and QA codes, the parameter always satisfies $T_\eta^* < L(\mathbf{U})/2$ for a signal to noise ratio greater than or equal to $0\,\text{dB}$. The same conclusion can be drawn for the set of VLC presented in [7].

C. Simulation results

The proposed algorithm has been applied to different VLC and QA codes. Its performance is compared against the ones of:

- A BCJR algorithm run on the aggregated model with parameter T_η^* and computing a bit posterior marginal only. The sequence of bits is estimated by taking a hard decision on the APP measures computed by the BCJR algorithm, and then a standard VLC/QA decoder is run to obtain the symbol sequence.
- A Viterbi algorithm run on the aggregated model.

In this section, the results are given in terms of SER, computed using the Hamming distance. The reference length use to compute the SER is taken as the one of the original sequence. Possibly missing or additional symbols in the estimated sequence are considered as wrong.

Fig. 2 shows the SER performance of the code $\mathcal{Q}_1$ for a range of channel signal to noise ratios E_b/N_0 for the three decoding schemes. The value η has been chosen equal to 10^{-6}, and the parameter T_η^* has been adjusted accordingly. The length of the symbol sequences considered in the simulations is equal to $L(\mathbf{U}) = 300$. The performance of the optimal BCJR that computes symbol posterior marginals on the full state model is not depicted on this graph as it is the same as the performance of the proposed algorithm. Note also that the performance obtained with the Viterbi algorithm corresponds to the first step of the proposed algorithm. The values of the parameter T_η^* used in the simulations and corresponding to $\eta = 10^{-6}$ lie in the integer interval $[25, 141]$.

Table I gives the SER performance of two VLC (denoted $\mathcal{C}_1$ and $\mathcal{C}_2$) and a QA code ($\mathcal{Q}_2$) for the three decoding techniques. The two VLC are defined for a 5-symbol alphabet of probabilities $\{0.4, 0.2, 0.2, 0.1, 0.1\}$. The set of codewords of $\mathcal{C}_1$ and $\mathcal{C}_2$ are $\{0, 11, 101, 1000, 1001\}$ and $\{01, 00, 11, 000, 001\}$, respectively. These two codes have the same mean description length. The QA code Q_2 is defined by the initial interval $[0, 8[$ for a binary alphabet with $\mathbb{P}(a_1) = 0.9$. When $\eta = 10^{-6}$, the parameter T_η^* is *optimal*, i.e. the FER and BER performance of the four codes on the aggregated model with parameter T_η^* are the same as the ones obtained with the bit/symbol model (cf [7] [8]). The length of the symbol sequences considered in the simulations of Table I is $L(\mathbf{U}) = 100$. The values of T_η^* used to obtain these results are also reported in this table.

The impact of the parameter T of the aggregated model on the SER performance, when this parameter is not *optimal* is shown in Fig. 3. Fig. 3 shows the SER performance of code $\mathcal{C}_1$ versus the aggregation parameter T for $E_b/N_0 = 5\,\text{dB}$ and the three decoding techniques of Fig. 2. The different values of T correspond to $\eta \in [10^{-6}, 10^{-1}]$. This figure depicts the impact of η (similarly T_η^*) on the performance of the proposed algorithm. Lower is η, more accurate is the estimation of the symbol clock from the modulo values of the aggregated model (first step of the algorithm), and hence lower is the SER after the second step of the algorithm.

The average reliability values output by the symbol-based BCJR algorithm, the Soft-output Viterbi Algorithm (SOVA) [15] [16] and the proposed decoding scheme for $\eta = 10^{-6}$ are reported in Table II for code $\mathcal{Q}_1$. These reliability values correspond to the values $\mathbb{P}(U_k \,|\, Y_1^{L(X)})$ output by the three different algorithms for each transmitted symbols averaged over 10^6 symbols. The quality of the soft outputs provided by the BCJR algorithm is *optimal*. We can see that the soft outputs provided by the 2-step algorithm closely approaches these optimal values. The quality of soft outputs provided by the SOVA is lower than the two previous algorithms.

IV. Conclusion

In this paper, we have proposed a two-step decoding algorithm that computes symbol a posteriori probabilities on

TABLE I

SER PERFORMANCE OF CODES $\mathcal{C}_1$, $\mathcal{C}_2$ AND $\mathcal{Q}_2$ VERSUS THE SIGNAL TO NOISE RATIO FOR DIFFERENT SOFT DECODING TECHNIQUES.

$E_b/N_0(dB)$	0	1	2	3	4	5	6	7
	Code $\mathcal{C}_1$							
Viterbi	0.5896460	0.5394457	0.4613242	0.3393017	0.1829289	0.0615358	0.0124061	0.0016876
Bit-based BCJR	0.6521376	0.6221372	0.5681771	0.4567565	0.2593654	0.0828331	0.0148831	0.0017665
2-step algorithm	0.5096099	0.4722298	0.4129241	0.3130139	0.1741670	0.0597935	0.0123041	0.0016345
T_η^*	43	39	34	30	25	21	16	12
	Code $\mathcal{C}_2$							
Viterbi	0.3572242	0.2781811	0.1944515	0.1161866	0.057206	0.0228583	0.0074904	0.0020121
Bit-based BCJR	0.4198277	0.336879	0.2382734	0.1409759	0.0666855	0.0250575	0.0076668	0.0020160
2-step algorithm	0.3324591	0.2620679	0.1851243	0.1125048	0.0563842	0.0227737	0.0074707	0.0020085
T_η^*	15	12	10	9	8	6	5	4
	Code $\mathcal{Q}_2$							
Viterbi	0.1466013	0.1326017	0.1103956	0.0773081	0.0397315	0.0140524	0.0030577	0.0004332
Bit-based BCJR	0.1628432	0.1531363	0.1353369	0.1016360	0.0545963	0.0175466	0.0034213	0.0004613
2-step algorithm	0.1035528	0.0931538	0.0848605	0.0674245	0.0380343	0.0133155	0.0029066	0.0004297
T_η^*	49	44	39	33	28	22	17	11

TABLE II

AVERAGE RELIABILITY VALUES (IN PROBABILITY) OUTPUT BY THREE DIFFERENT ALGORITHMS

$E_b/N_0(dB)$	0	1	2	3	4	5	6	7
	Code $\mathcal{Q}_1$							
SOVA	0.7006647	0.7372412	0.7814617	0.8303492	0.8756119	0.9088100	0.9291074	0.9415482
2-step algorithm	0.7204538	0.7615880	0.8142597	0.8761704	0.9352687	0.9755301	0.9934656	0.9987168
Symbol-based BCJR	0.7204547	0.7616008	0.8142599	0.8762152	0.9353449	0.9755583	0.9935396	0.9987495

an aggregated state model for VLC or QA codes. The two steps of the proposed algorithm can be simultaneously performed to improve the efficiency of this algorithm in terms of computational cost. This algorithm yields SER performance close to the ones obtained on the optimal (complete) model of high complexity, provided that the parameter of the aggregated model is appropriately chosen. This APP can be used in iterative decoding schemes in order to minimize the SER.

REFERENCES

[1] V. B. Balakirsky, "Joint source-channel coding with variable length codes," in *Proc. Intl. Conf. Inform. Theory, ISIT*, 1997, p.419.

[2] C. Weidmann, "Reduced-complexity soft-in-soft-out decoding of variable length codes," in *Proc. Intl. Conf. Inform. Theory, ISIT*, Yokohama, Japan, July 2003.

[3] X. Jaspar and L. Vandendorpe, "New iterative decoding of variable length codes with turbo codes," in *Proc. Intl. Conf. Commun., ICC*, Paris, France, June 2004.

[4] R. Bauer and J. Hagenauer, "Symbol by symbol map decoding of variable length codes," in *Proc. 3rd ITG Conf. Source and Channel Coding, Munich, Germany, Jan. 17.-19. 2000*, 2000. [Online]. Available: citeseer.nj.nec.com/bauer00symbolbysymbol.html

[5] T. Guionnet and C. Guillemot, "Soft and joint source-channel decoding of quasi-arithmetic codes," *EURASIP Journal on applied signal processing*, vol. 3, pp. 394–411, Mar. 2004.

[6] S. Ben-Jamaa, C. Weidmann, and M. Kieffer, "Asymptotic error-correcting performance of joint source-channel schemes based on arithmetic coding," in *Proc. MMSP*, Victoria, Canada, October 2006, pp. 262–266.

[7] S. Malinowski, H. Jégou, and C. Guillemot, "Synchronization recovery and state model reduction for soft decoding of variable length codes," *IEEE Trans. Inform. Theory*, pp. 368–377, Jan. 2007.

[8] S. Malinowski, H. Jégou, and C.Guillemot, "Error recovery properties of quasi-arithmetic codes and soft decoding with length constraint," in *Proc. Intl. Conf. Inform. Theory, ISIT*, Seattle, USA, July 2006.

[9] J. Kliewer and R. Thobaben, "Combining FEC and optimal soft-input source decoding for the reliable transmission of correlated variable-length encoded signals," in *Proc. Data Compression Conf., DCC*, April 2002, pp. 83–91.

[10] L. R. Bahl, J. Cocke, F. Jelinek, and J. Raviv, "Optimal decoding of linear codes for minimizing symbol error rate," *IEEE Trans. Inform. Theory*, pp. 284–287, Mar. 1974.

[11] J. Rissanen, "Generalized Kraft inequality and arithmetic coding," *IBM Jl. Res. Develop.*, vol. 20, pp. 198–203, 1976.

[12] G. Langdon, "An introduction to arithmetic coding," *IBM Jl. Res. Develop.*, vol. 28, 1984.

[13] D. Bi, M. Hoffman, and K. Sayood, "State machine interpretation of arithmetic codes for joint source and channel coding," in *Proc. Data Compression Conf., DCC*, Mar. 2006, pp. 143–152.

[14] A. Viterbi, "Error bounds for convolution codes and an asymptotically optimum decoding algorithm," *IEEE Trans. Inform. Theory*, no. 13, pp. 260–269, 1967.

[15] G. Battail, "Pondération des symboles décodés par l'algorithme de Viterbi," *Ann. Telecommmun.*, vol. 42, pp. 31–38, Jan. 1987.

[16] L. Lin and R. Cheng, "Improvements in SOVA-based decoding for turbo codes," in *Proc. ICC'97*, Montreal, Canada, June 1997.